WEB PHISHING DETECTION

IBM-Project-5336-1658758088

NALAIYATHIRAN PROJECT BASED ON LEARNING PROFESSIONAL READLINESS FOR INNOVATION, EMPLOYMENT AND ENTERPRENEURSHIP

Project Report

TEAM ID: PNT2022TMID47507

TEAM MEMBERS:

1.GAYATHRI.A – 910419104002

2.SANTHIYA.S – 910419104019

3.SHALLY THERSE.P – 910419104020

4.MUTHUPRIYA.G – 910419104013

5.SURUTHI.S - 910419104023

BACHELOR OF ENGINEERING IN COMPUTER SCIENCE AND ENGINEERING

FATIMA MICHAEL COLLEGE OF ENGINEERING AND ECHNOLOGY
MADURAI – 625 020

INDEX

1. INTRODUCTION

1.1 Project Overview

This project mainly focuses on applying a machine-learning algorithm to detect phishing websites. In order to detect and predict phishing websites, we proposed an intelligent, flexible, and effective system that is based on using classification algorithms. We implemented classification algorithms and techniques to extract the phishing dataset's criteria to classify their legitimacy. The phishing website can be detected based on some important characteristics, like the URL and domain identity, and security and encryption criteria in the final phishing detection rate. Once a user enters a website, our system will use a data mining algorithm to detect whether the website is a phishing website or not.

1.2 Purpose

There are a number of users who purchase products online and make payments through e-banking. Some e-banking websites ask users to provide sensitive data such as username, password, and credit card details, etc., often for malicious reasons. This type of e-banking website is known as a phishing website. Web services are one of the key communications software services for the Internet. Web phishing is one of many security threats to web services on the Internet. There are millions of incidents happening around the world in an hour. People suffer immeasurable losses due to these attacks. Therefore, protecting users from such attacks is the sole purpose of our project.

The simplest method of obtaining sensitive information from unwitting users is through phishing attacks. The goal of phishers is to obtain vital data, such asusername, password, and bank account information. People working in cyber security are currently searching for reliable and consistent methods of detecting phishing websites. In this research, many properties of legal and phishing URLs are extracted and analyzed in order to detect phishing URLs. The algorithms used to identify phishing websites include decision trees, random forests, and support vector machines. By evaluating each algorithm's accuracy rate, false positive rate, and falsenegative rate, the study aims to identify phishing URLs as well as identify the best machine learning method.

2. LITERATURE SURVEY

2.1 Existing problem

Due to how simple it is to create a fake website that closely resembles a legitimate website, phishing has recently become a top concern for security researchers. Experts can spot fake websites, but not all users can, and those users end up falling for phishing scams. The attacker's primary goal is to steal bank account credentials. Businesses in the US lose $2 billion annually as a result of their customers falling for phishing scams. The annual global impact of phishing was estimated to be as high as $5 billion in the third Microsoft Computing Safer Index Report, which was published in February 2014. Because users are unaware of phishing attacks, they are becoming more successful.

Since phishing attacks take advantage of user vulnerabilities, it is highly challenging to counteract them, but it is crucial to improve phishing detection methods. The common technique, commonly referred to as the "blacklist" method, for detecting phishing websites involves adding Internet Protocol (IP) blacklisted URLs to the antivirus database. Attackers utilize clever methods to deceive people by changing the URL to seem authentic through obfuscation and many other straightforward tactics, such as fast-flux, in which proxies are automatically constructed to host the website, algorithmic production of new URLs, etc. This method's primary flaw is that it cannot identify phishing attacks that occur at zero hour.

Zero-hour phishing attacks can be detected using heuristic-based detection, which includes characteristics that have been observed to exist in phishing attacks in reality. However, the presence of these characteristics is not always guaranteed in such attacks, and the false positive rate for detection is very high.

2.2 References

S.NO	PAPER TITLE	PAPER CONCEPT	ADVANTAGE	DISADVANTAGE
1	LongfeiWu etal..., "**Effective Defense Schemes for Phishing Attacks on Mobile Computing Platforms**, " IEEE 2016, pp.6678-6691.	In this paper, author did a comprehensive study on the Security vulnerabilities caused by mobile phishing attacks, including the web page phishing attacks.	Author propose MobiFish, a novel automated lightweight anti- phishing scheme for mobile platforms. MobiFish verifies the validity of web pages, applications, and persistent accounts by comparing the actual Identity to the claimed identity	Existing schemes designed for web phishing attacks on PCs **c**annot effectively address the various phishing attacks on mobile devices.
2	Surbhi Gupta etal., "**A Literature Survey on Social Engineering Attacks: Phishing Attacks**," in International Conference on Computing, Communication and Automation(ICCCA2016),2016, pp. 537-540.	To fool an online user into elicit personal Information. The prime objective of this review is to do literature survey on social engineering attack: Phishing attacks and techniques to detect attack.	The paper discusses various types of Phishing attacks such as Tab-napping, spoofing emails, Trojan horse, hacking and how to prevent them.	Every organization has security issues that have been of great concern to u sets, sited developers, and specialists, in order to defend the confidential data from this type of social engineering attack.

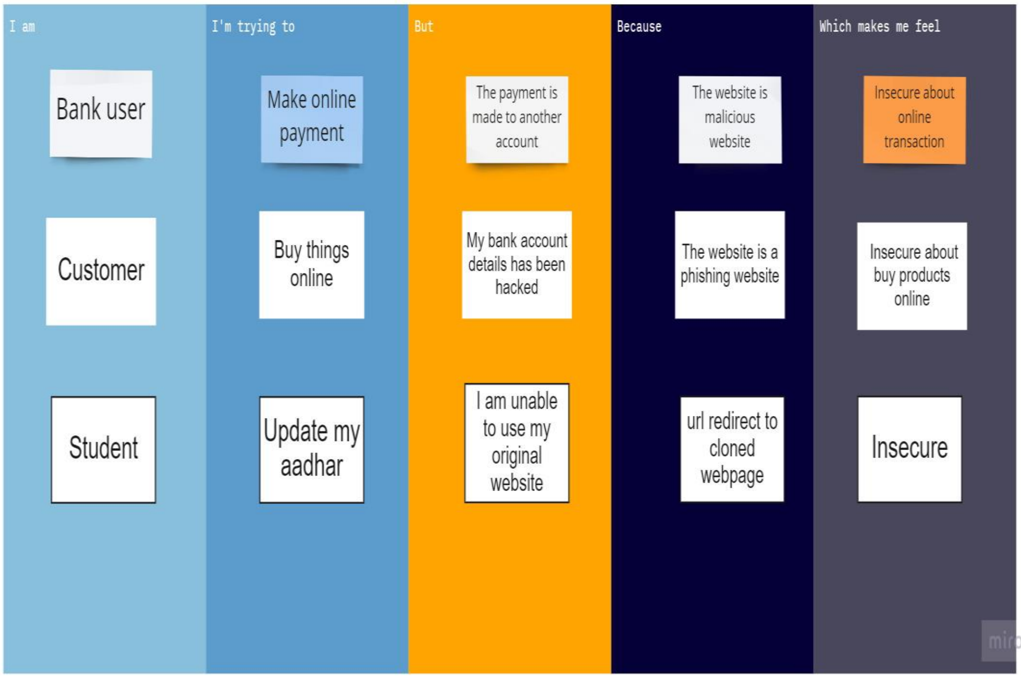
3	Guardian Analytics, "**A Practical Guide to Anomaly Detection Implications of meeting new FFIEC minimum expectations for layered security**". [Accessed : 08 Jan 2015]	Commercial and retail account holders at financial institutions of all sizes are under attacks by sophisticated, Organized, Well-funded cyber criminals.	Anomaly detection solutions are readily available, are deployed quickly and immediately and automatically protect all account holders against all types of fraud attack with minimal Disruption to legitimate online banking activity.	Implementing anomaly detection will not only meet FFIEC expectations, it will decrease the total cost of fraud, and will increase customer loyalty and trust.
4	SANS Institute, "**Phishing : Analysis of a Growing Problem**",2007. 1417[Accessed : 23 May 2017]	This paper gives an in depth analysis of phishing: what it is, the technologies and security. Weaknesses it takes advantage of the dangers it poses to end users.	In this analysis author explain the concepts and technology behind phishing, show how the threat is much more than just a nuisance or passing trend, and discuss how gangs of criminals are using these scams to make a great deal of money.	Unfortunately, a growing number of cyber-thieves are using these same systems to manipulate us and steal our private information.

5	J. Phys.: Conf. Ser. "**A literature survey on Retraction: Phishing website detection using machine learning and deep learning techniques**" 1916 (2021) 012407.	Nowadays, website phishing is more damaging. It is becoming a big threat to people's daily life and networking environment. In these attacks, the intruder puts on an act as if it is a trusted organization with an intention to purloin liable and essential information. The methodology we discovered is a powerful technique to detect the phished websites and can provide more effective defenses for phishing attacks of the future.	The association between independent variables as well as dependent variables can be formed without any presumptions about the statistical depiction of the aspect. It contributes positive gains on regression algorithm which includes its competence to act with noisy data.	The ANN's are not suitable for infrequent or utmost events where data is inadequate in order to train it. ANNs do not permit the embodiment of human mastery to be substitutive for perceptible proof.
6	**"Phishing Website Detection Based on Deep Convolutional Neural Network and Random**	This paper proposes an integrated phishing website detection method based on convolutional neural networks	A 99.35% correct classification rate of phishing websites was obtained on the dataset. Experiments were conducted on the test set and training set, and the	It takes longer to train. However, the trained model is better than the others in terms of accuracy of phishing website detection. Another disadvantage is

	Forest Ensemble Learning" ,This research was funded by the National Key R & D Program of China Grant Numbers 2017YFB0802800 and Beijing Natural Science Foundation (4202002)	(CNN) and random forest (RF). The method can predict the legitimacy of URLs without accessing the web content or using third-party services. The proposed technique uses **c**haracter embedding techniques to convert URLs into fixed-size matrices, extract features at different levels using CNN models, classify multi-level features using multiple RF classifiers, and, finally, output prediction results using a winner-take-all approach.	experimental results proved that the proposed method has good generalization ability and is useful in practical applications.	that the model cannot determine whether the URL is active or not, so it is necessary to test whether the URL is active or not before detection to ensure the effectiveness of detection. In addition, some attackers use URLs that are not imitations of other websites, and such URLs will not be detected.

2.3 Problem Statement Definition

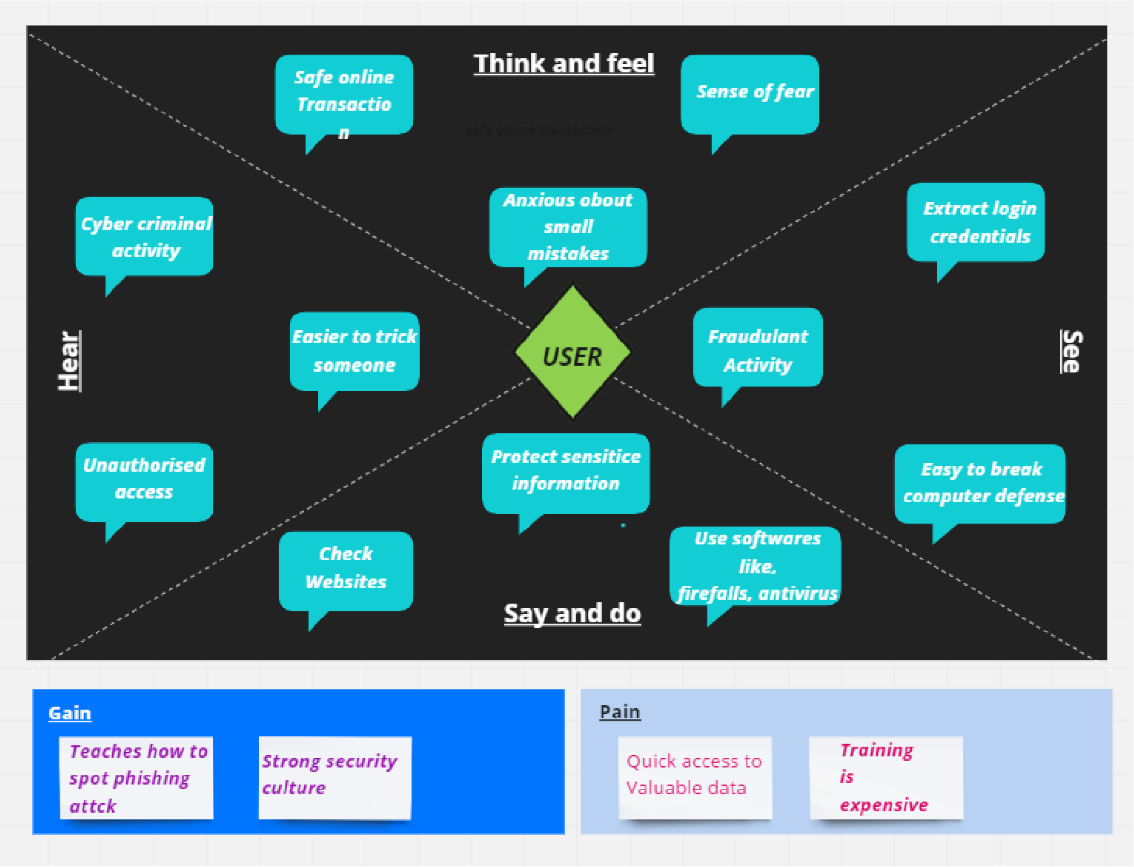
Human users' inability to recognize phishing sites allows phishing attacks to succeed. Past work in anti-phishing can be broadly divided into four categories: studies to understand why people fall for phishing attacks, strategies for teaching people not to fall for phishing attacks, user interfaces for assisting people in making better decisions about trusting email and websites, and automated tools to detect phishing. Our research outlines a method for automatically identifying phishing. Most end users typically base their decisions only on how they feel and how they look. When a user accesses the internet, all they see is a browser's screen. After that, he or she works on a web page's command. Most phishing efforts take use of this sort of unintended chance provided by the user and trick them since the user is unconcerned with the back end procedure.



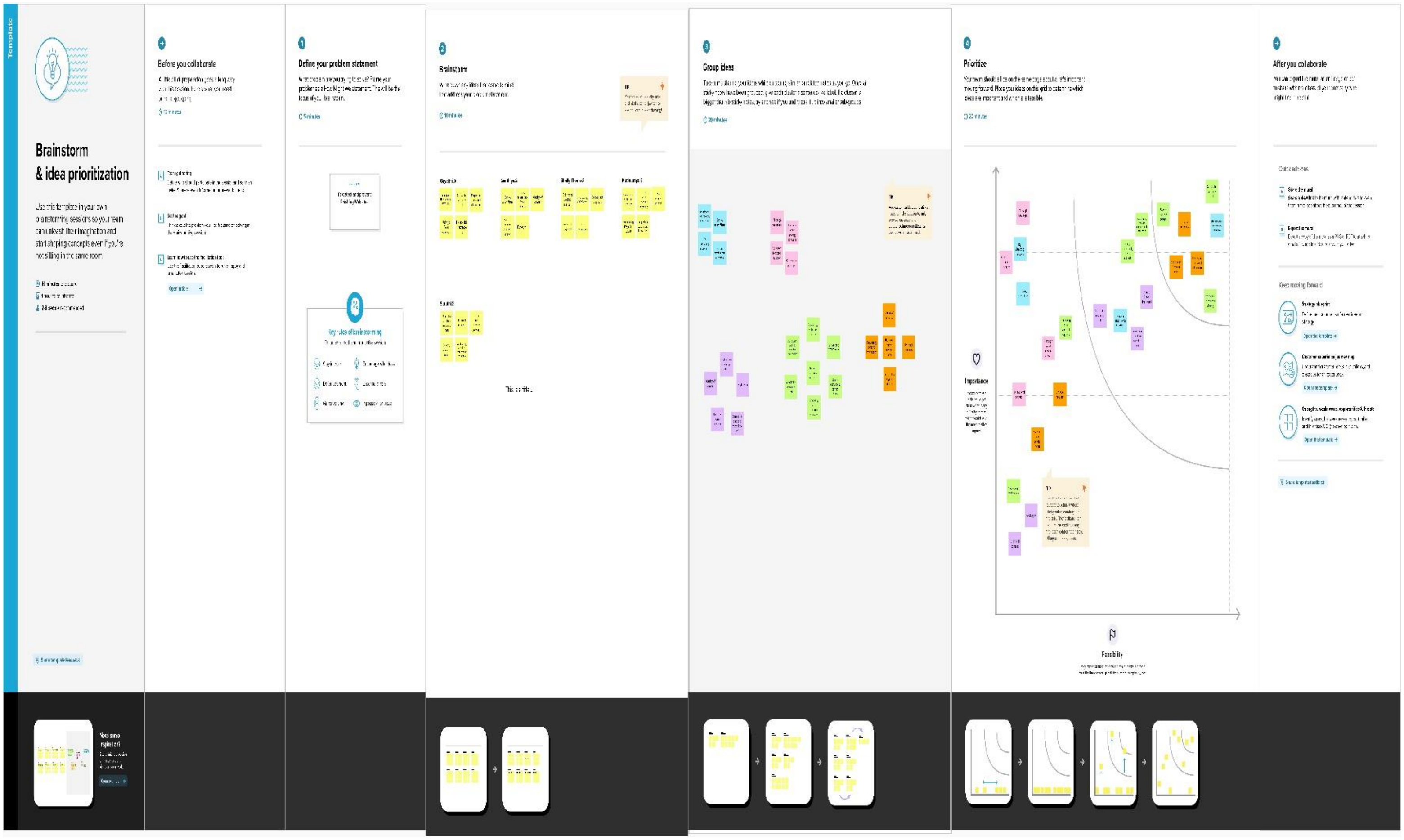
I am	I'm trying to	But	Because	Which makes me feel
Bank user	Make online payment	The payment is made to another account	The website is malicious website	Insecure about online transaction
Customer	Buy things online	My bank account details has been hacked	The website is a phishing website	Insecure about buy products online
Student	Update my aadhar	I am unable to use my original website	url redirect to cloned webpage	Insecure

3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas



3.2 Ideation & Brainstorming



Template

Brainstorm
& idea prioritization

Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

Before you collaborate

Define your problem statement

5 minutes

Key rules of brainstorming

Brainstorm

10 minutes

Group ideas

20 minutes

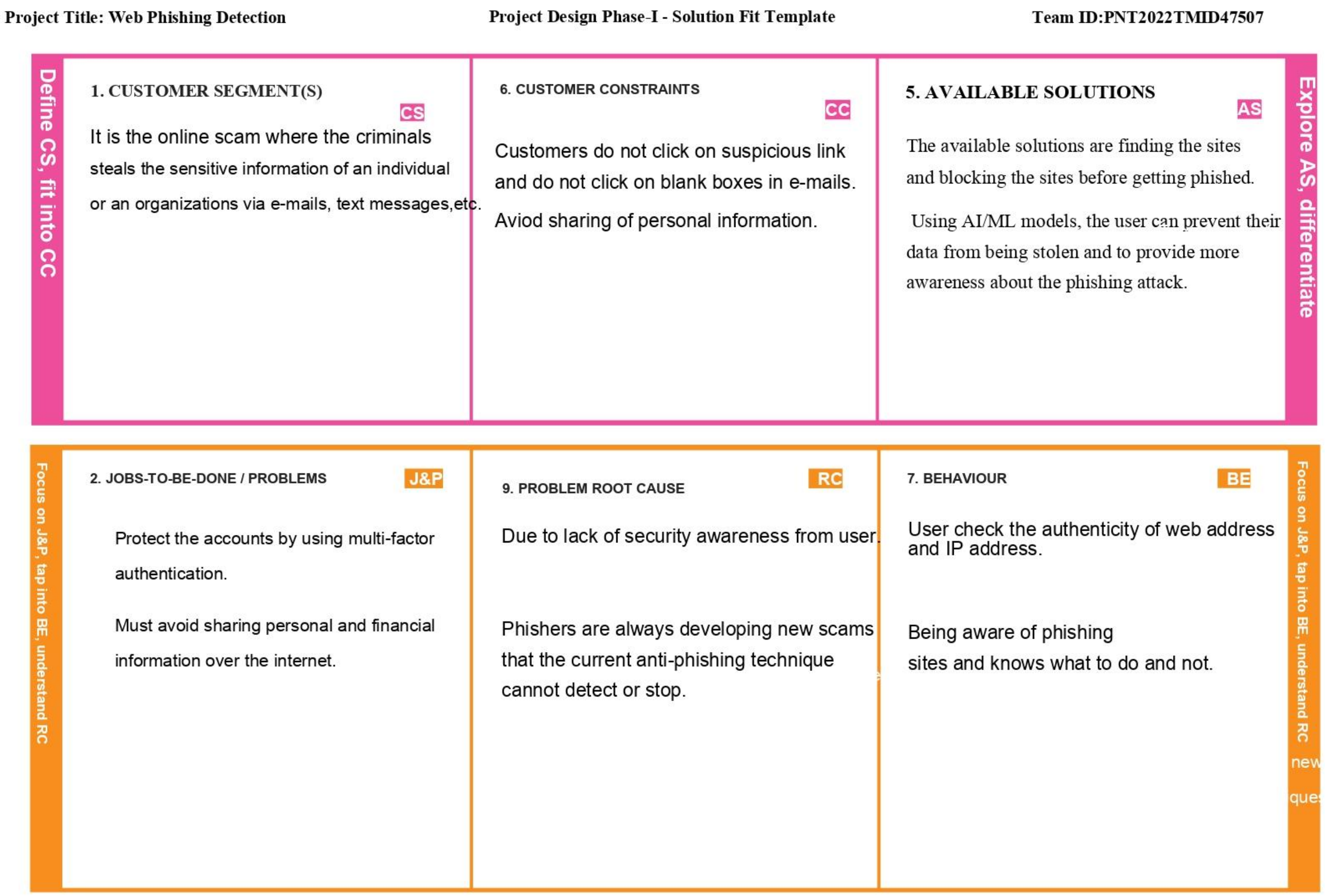
Prioritize

After you collaborate

3.3 Proposed Solution

S.No	Parameter	Description
1.	Problem Statement (problem to besolved)	To detect whether the e-banking website is Phishing Website or not.
2.	Idea/ Solution description	By using Machine Learning we are going to detect the e-banking websitesand it's detection based on some important characteristics such as URL, domain identity and security.
3.	Novelty/Uniqueness	We implemented classification algorithms and techniques to extractthe phishing datasets criteria to classify their legitimacy.
4.	Social Impact / Customer Satisfaction	Preventing our credentials details and to safeguard online users from becoming victims of online frauds, andalso against many other hacking.
5.	Business Model (Revenue Model)	By detecting the phishing websites in business, it prevents productivity, dataloss and reputational damage.
6.	Scalability of the Solution	To incorporate security awareness into the organization and to protect user's credentials or sensitive data.

3.4 Problem Solution fit



Project Title: Web Phishing Detection **Project Design Phase-I - Solution Fit Template** **Team ID:PNT2022TMID47507**

Define CS, fit into CC

1. CUSTOMER SEGMENT(S) CS

It is the online scam where the criminals steals the sensitive information of an individual or an organizations via e-mails, text messages,etc.

6. CUSTOMER CONSTRAINTS CC

Customers do not click on suspicious link and do not click on blank boxes in e-mails.

Aviod sharing of personal information.

5. AVAILABLE SOLUTIONS AS

The available solutions are finding the sites and blocking the sites before getting phished.

Using AI/ML models, the user can prevent their data from being stolen and to provide more awareness about the phishing attack.

Explore AS, differentiate

Focus on J&P, tap into BE, understand RC

2. JOBS-TO-BE-DONE / PROBLEMS J&P

Protect the accounts by using multi-factor authentication.

Must avoid sharing personal and financial information over the internet.

9. PROBLEM ROOT CAUSE RC

Due to lack of security awareness from user.

Phishers are always developing new scams that the current anti-phishing technique cannot detect or stop.

7. BEHAVIOUR BE

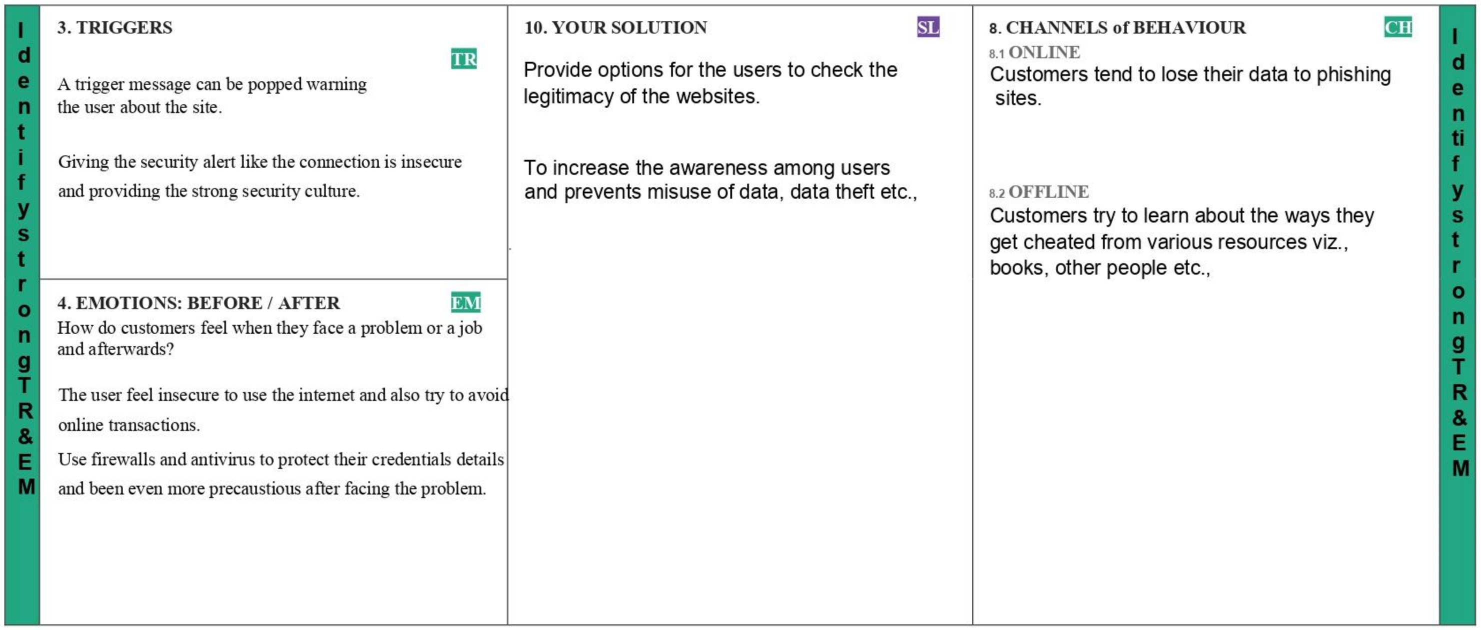
User check the authenticity of web address and IP address.

Being aware of phishing sites and knows what to do and not.

Focus on J&P, tap into BE, understand RC

new

ques



Identify strong TR&EM

3. TRIGGERS TR

A trigger message can be popped warning the user about the site.

Giving the security alert like the connection is insecure and providing the strong security culture.

4. EMOTIONS: BEFORE / AFTER EM

How do customers feel when they face a problem or a job and afterwards?

The user feel insecure to use the internet and also try to avoid online transactions.

Use firewalls and antivirus to protect their credentials details and been even more precaustious after facing the problem.

10. YOUR SOLUTION SL

Provide options for the users to check the legitimacy of the websites.

To increase the awareness among users and prevents misuse of data, data theft etc.,

8. CHANNELS of BEHAVIOUR CH

8.1 ONLINE

Customers tend to lose their data to phishing sites.

8.2 OFFLINE

Customers try to learn about the ways they get cheated from various resources viz., books, other people etc.,

Identify strong TR&EM

4. REQUIREMENT ANALYSIS

4.1 Functional requirement:

Following are the functional requirements of the proposed solution.

A functional of software system is defined in functional requirementsand the behaviour of the system is evaluated when presented with specific inputs or conditions which may include calculations, data manipulation and processing and other specific functionality.

- Our system should be able to load air quality data and pre-process data.
- It should be able to analyse the air quality data.
- It should be able to group data based on hidden patterns.
- It should be able to assign a label based on its data groups.
- It should be able to split data into trainset and testset.
- It should be able to train model using trainset.
- It must validate trained model using testset.
- It should be able to display the trained model accuracy.
- It should be able to accurately predict the air on unseen data.

4.2 Non-Functional requirements:

Following are the non-functional requirements of the proposed solution.

ACCESSIBILITY:

Availability is a general term used to depict how much an item, gadget, administration, or condition is open by however many individuals as would be prudent.

In our venture individuals who have enrolled with the cloud can get tothe cloud to store and recover their information with the assistance of a mystery key sent to their email ids.

UI is straightforward and productive and simple to utilize.

MAINTAINABILITY:

In programming designing, viability is the simplicity with which aproduct item can be altered so as to:

- Correct absconds
- Meet new necessities

New functionalities can be included in the task based the client necessities just by adding the proper documents to existing ventureutilizing ASP.net and C# programming dialects. Since the writing computer programs is extremely straight-forward, it is simpler to discover and address the imperfections and to roll out the improvements in the undertaking.

SCALABILITY:

Framework is fit for taking care of increment all out throughput underan expanded burden when assets (commonly equipment) are included.

Framework can work ordinarily under circumstances, for example,low data transfer capacity and substantial number of clients.

PORTABILITY:

Convey ability is one of the key ideas of abnormal state programming. Convenient is the product code base components to have the capacity to reuse the current code as opposed to making newcode while moving programming from a domain to another. Venture can be executed under various activity conditions gave it meet its basesetups. Just framework records and dependant congregations would need to be designed in such case.

The functional requirements for a system describe what the systemshould do.

Those requirements depend on the type of software being developed, the expected users of the software. These are the statement of services the system should provide, how the system should react to particular inputs and how the system should behave in particular situation.

1. Extracting data from csv files
2. Cleaning the data
3. Vector representation

Non-functional requirements is not about functionality or behaviourof system, but rather are used to specify the capacity of a system.
They are more related to properties of system such as quality, reliability and quick response time. Non-functional requirements come up via customer needs, because of budget, interoperability need such as software and hardware requirement, organisational policies ordue to some external factors such as:-

- Basic operational Requirement
- Organisational Requirement
- Product Requirement
- User Requirement

HARDWARE REQUIREMENTS:

The following is the hardware requirements of the system for theproposed system:

- Processor: Any Processor above 500MHZ
- RAM: 8 GB
- Hard Disk: 1TB
- Input device: Standard keyboard and mouse.

SOFTWARE REQUIREMENTS:

The following is the software requirements of the system for theproposed system:

OS	: Windows 10
Platform	: Jupyter Notebook
Language	: Python
IDE/tool	: Anaconda 3-5.0.3

SUPPORTING PYTHON MODULES

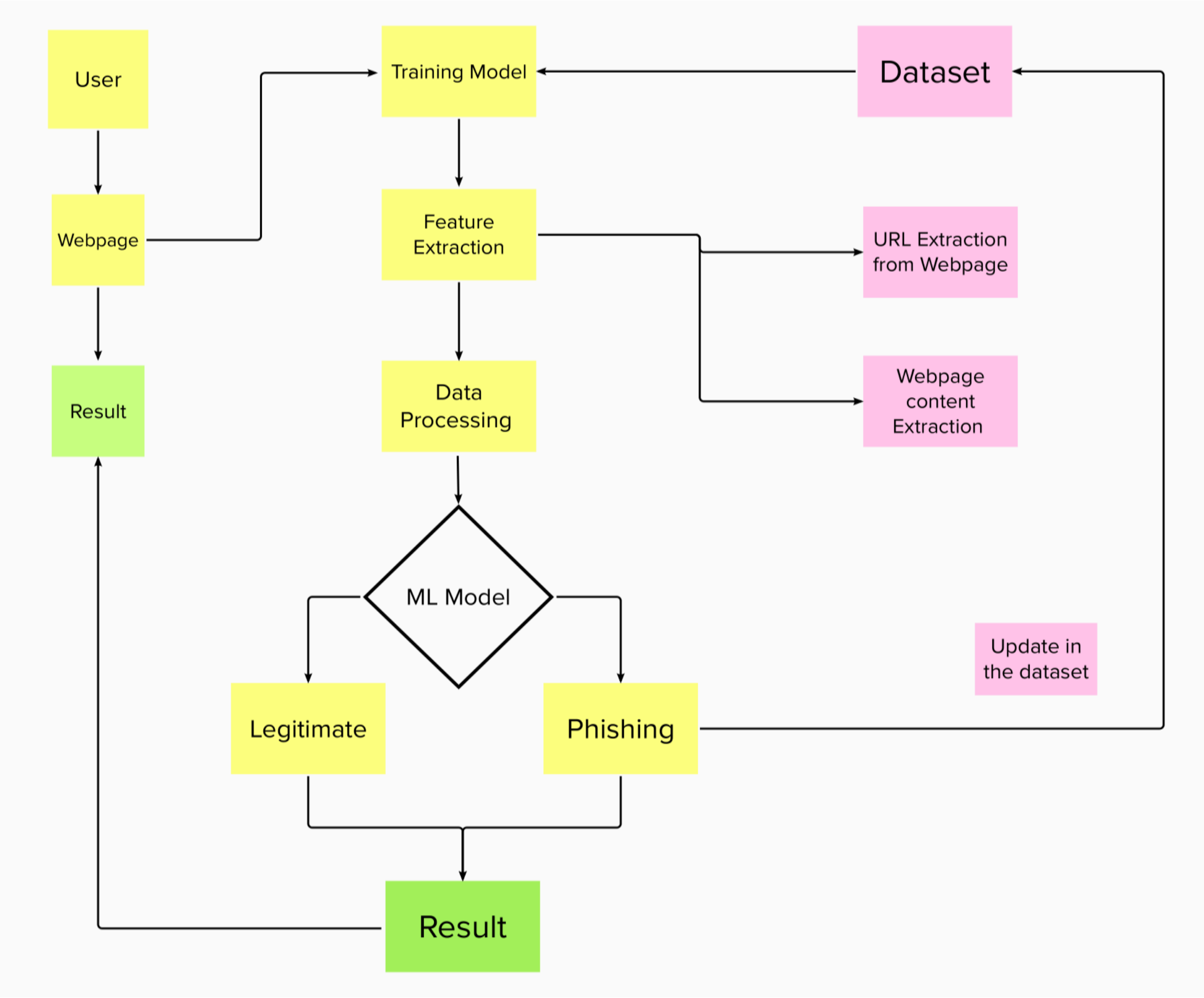
Python has an approach to place definitions in a document or in an intuitive case of the interpreter. Such a file is known as a module; definitions from a modules can be brought into different modules orinto the fundamental module. Some of the modules used in the project.

S.no	Python Module	Description
1	Ip address	Ip address gives the capacities to generate, control and work on IPv4 and IPv6 addresses and networks.
2	Whois	WHOIS is an inquiry and response convention that is comprehensively used for addressing database that store the selected customers or trustees of an internet resource. for example, a domain name, an autonomous framework or an IP address block, also simultaneously used for broad extend of an information.

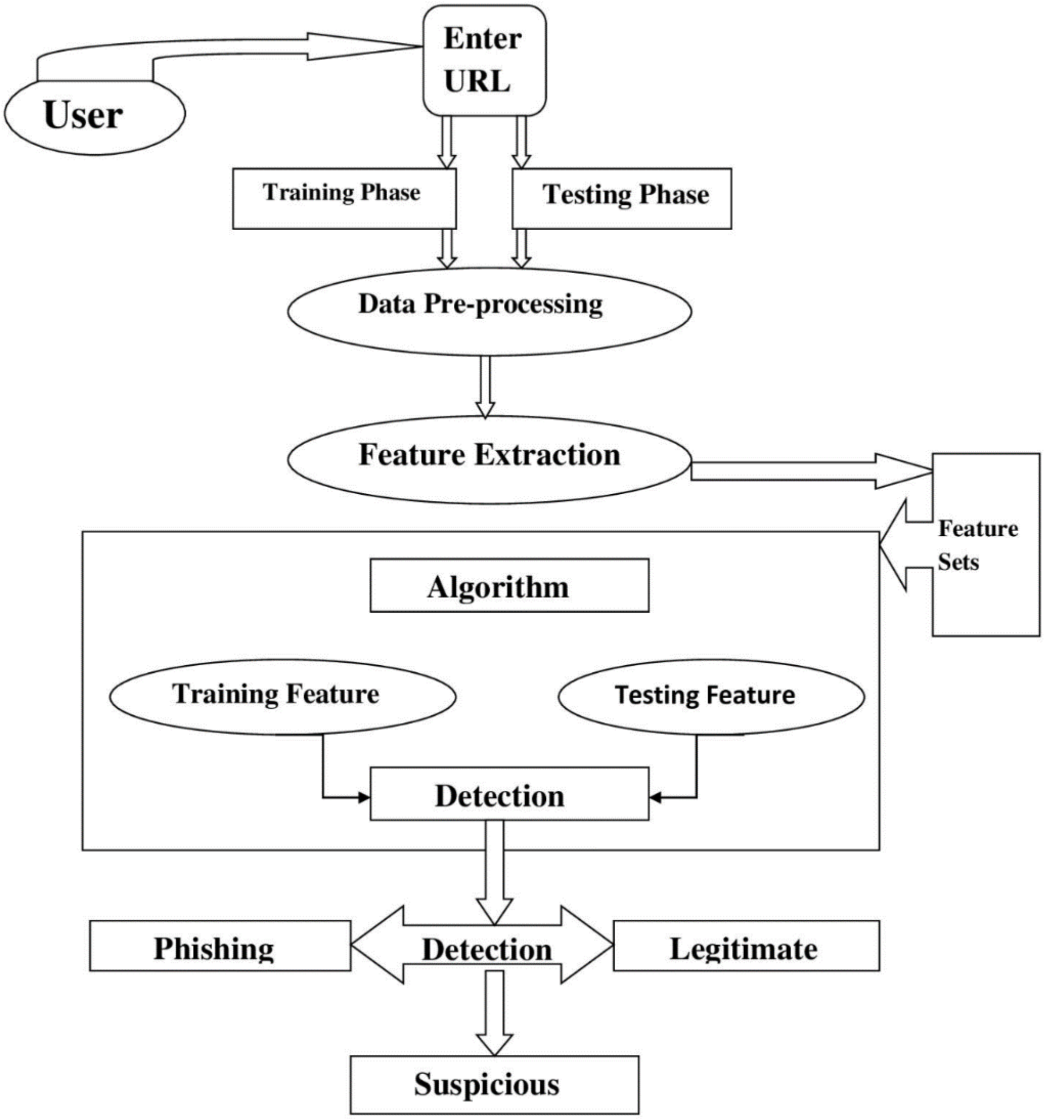
3	Re	This module gives regular expression matching activities like those found in Perl.
4	urllib.request	The urllib.request module characterizes functions and classes which help in opening URLs (for the most part HTTP) in a complex world.
5	Beautiful Soup	Beautiful Soup is a package in python for parsing HTML and XML records. It makes a parse tree for parsed pages that can be utilized to extricate information from HTML, which is valuable for web scraping
6	Socket	The BSD interface of socket is given access by this module
7	Requests	The HTTP requests are allowed to send by this module making use of Python.

5 PROJECT DESIGN

5.2 Data Flow Diagrams:



5.3 Solution & Technical Architecture



5.4 User Stories:

User Type	Functional Requirement(Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority
Customer (Mobile user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account /dashboard	High
		USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High
		USN-3	As a user, I can register for the application through Facebook	I can register & access the dashboard with Facebook Login	Low
		USN-4	As a user, I can register for the application through Gmail		Medium
	Login	USN-5	As a user, I can log into the application by entering email & password		High
	Dashboard				
Customer (Webuser)	User input	USN-1	As a user i can input the particular URL in the required field and waiting for validation.	I can go access the websitewithout any problem	High
Customer Care Executive	Feature extraction	USN-1	After i compare in case if none found on comparison then we can extract feature using heuristic and visual similarity approach.	As a User i can have comparison between websites for security.	High
Administrator	Prediction	USN-1	Here the Model will predict the URL websites using Machine Learning algorithms such as Logistic Regression, KNN	In this i can have correct prediction on the particular algorithms	High
	Classifier	USN-2	Here i will send all the model output to classifier in order to produce final result.	I this i will find the correctclassifier for producing the result	Medium

6 PROJECT PLANNING & SCHEDULING

6.2 Sprint Planning & Estimation

Sprints	User Type	Functional Requirement (Epic)	User Story No	User Story / Task	Story points	Team members	Priority
Sprint-1	Dataset collection and preprocessing	Fetch electronic mail messages	USN-1	As a new user, I will register first.	35	Gayathri. A Santhiya. S Shally Therse. P Suruthi. S Muthupriya. G	High
Sprint-2	Model and application building	Extract URLs	USN-2	As a user, I will provide specific URL for checking	15	Gayathri. A Santhiya. S Shally Therse. P Suruthi. S Muthupriya. G	High
Sprint-3	Feature addition for prediction page	Extract Header Information	USN-3	As a user, I wait for the application to classify it based on certain criteria.	25	Gayathri. A Santhiya. S Shally Therse. P Suruthi. S Muthupriya. G	High
Sprint-4	User acceptance testing, performance testing, migration from mongo DB to DB2	Classify the website	USN-4	As a user, I will be informed whether the link is suspicious or safe to use	25	Gayathri. A Santhiya. S Shally Therse. P Suruthi. S Muthupriya. G	High

6.3 Sprint Delivery Schedule

Sprint	Total Story Points	Duration	Sprint StartDate	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	35	7Days	29-10-2022	5-11-2022	35	4-11-2022
Sprint-2	15	8 Days	7-11-2022	14-11-2022	15	13-11-2022
Sprint-3	25	8 Days	16-11-2022	23-11-2022	25	23-11-2022
Sprint-4	25	8 Days	23-11-2022	30-11-2022	25	30-11-2022

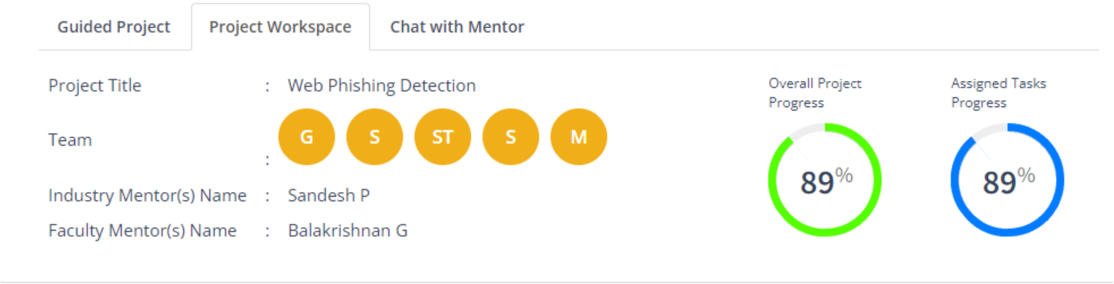
Velocity:

AV=Velocity/Duration = 35/7 =5

AV=Velocity/Duration = 15/8 =1.875A

V=Velocity/Duration = 25/8 =3.125

6.4 Reports from JIRA



Project Title : Web Phishing Detection

Team : G S ST S M

Industry Mentor(s) Name : Sandesh P

Faculty Mentor(s) Name : Balakrishnan G

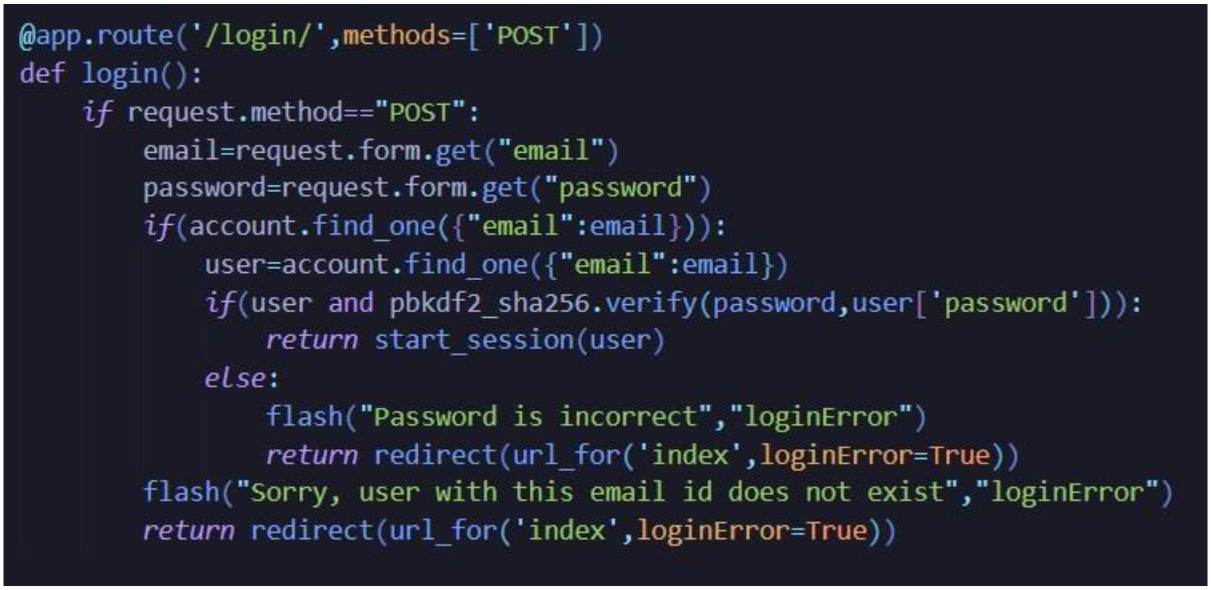
Overall Project Progress 89%

Assigned Tasks Progress 89%

7 CODING & SOLUTIONING (Explain the features added in theproject along with code)

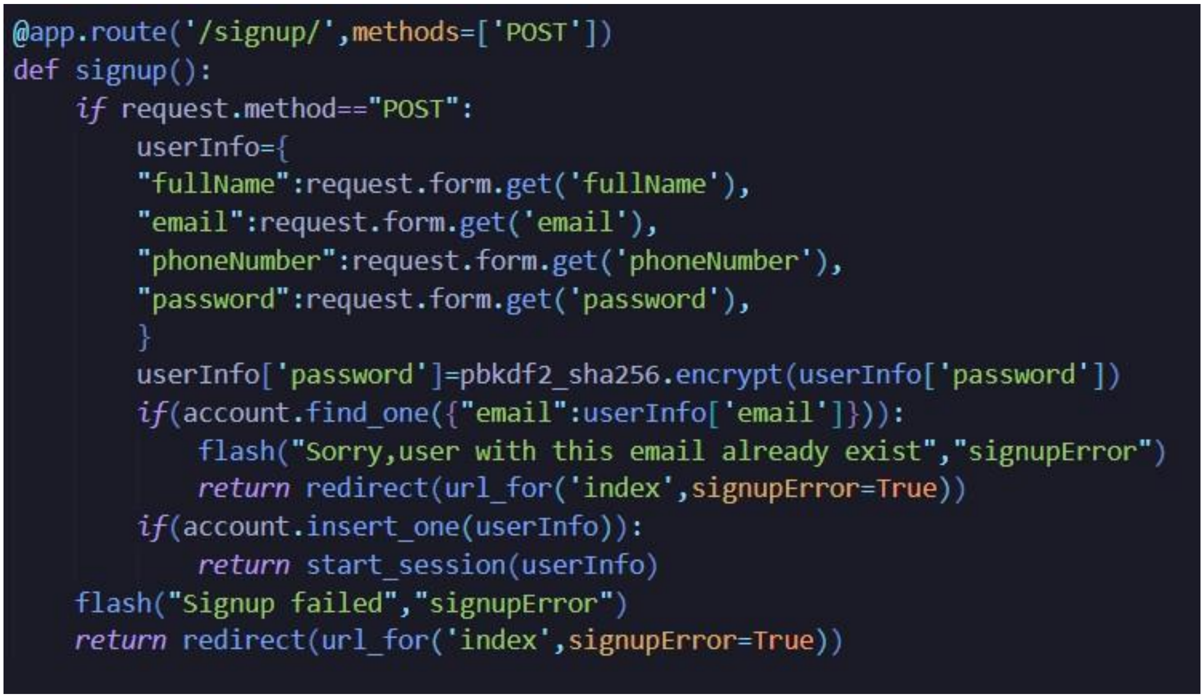
7.2 Feature 1

LOGIN



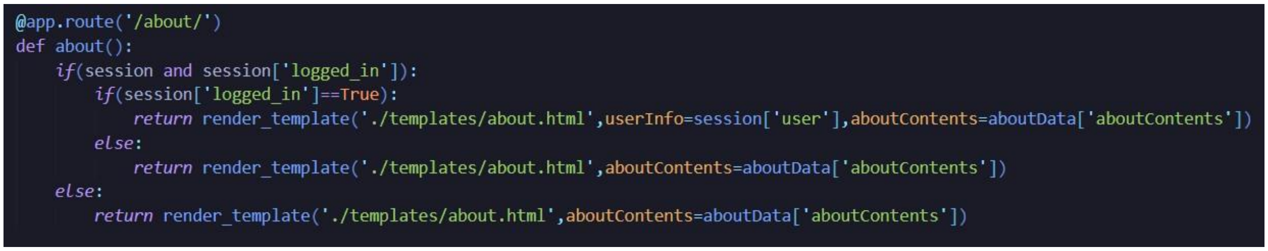
```
@app.route('/login/',methods=['POST'])
def login():
    if request.method=="POST":
        email=request.form.get("email")
        password=request.form.get("password")
        if(account.find_one({"email":email})):
            user=account.find_one({"email":email})
            if(user and pbkdf2_sha256.verify(password,user['password'])):
                return start_session(user)
            else:
                flash("Password is incorrect","loginError")
                return redirect(url_for('index',loginError=True))
        flash("Sorry, user with this email id does not exist","loginError")
        return redirect(url_for('index',loginError=True))
```

SIGNUP



```python
@app.route('/signup/',methods=['POST'])
def signup():
    if request.method=="POST":
        userInfo={
        "fullName":request.form.get('fullName'),
        "email":request.form.get('email'),
        "phoneNumber":request.form.get('phoneNumber'),
        "password":request.form.get('password'),
        }
        userInfo['password']=pbkdf2_sha256.encrypt(userInfo['password'])
        if(account.find_one({"email":userInfo['email']})):
            flash("Sorry,user with this email already exist","signupError")
            return redirect(url_for('index',signupError=True))
        if(account.insert_one(userInfo)):
            return start_session(userInfo)
    flash("Signup failed","signupError")
    return redirect(url_for('index',signupError=True))
```

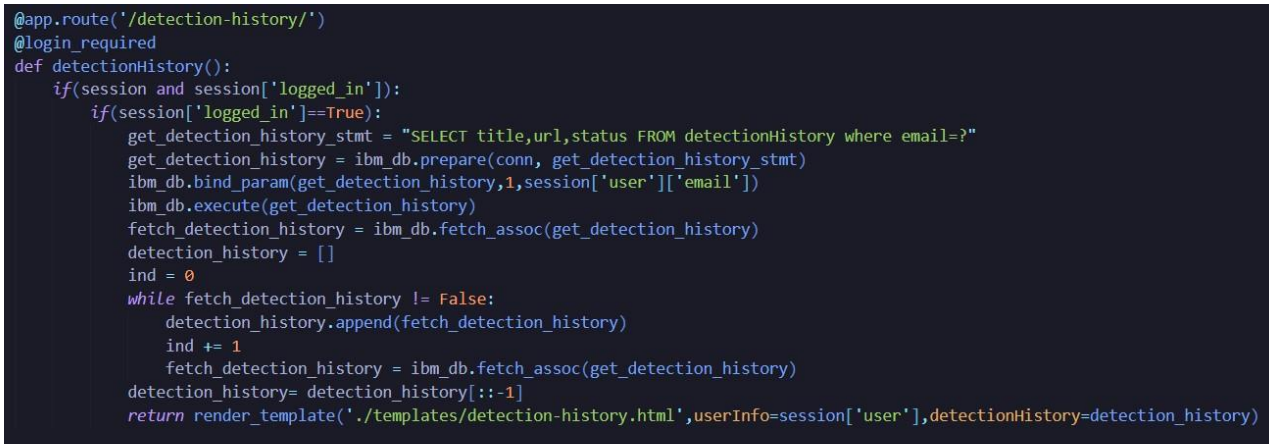
ABOUT US



```python
@app.route('/about/')
def about():
    if(session and session['logged_in']):
        if(session['logged_in']==True):
            return render_template('./templates/about.html',userInfo=session['user'],aboutContents=aboutData['aboutContents'])
        else:
            return render_template('./templates/about.html',aboutContents=aboutData['aboutContents'])
    else:
        return render_template('./templates/about.html',aboutContents=aboutData['aboutContents'])
```

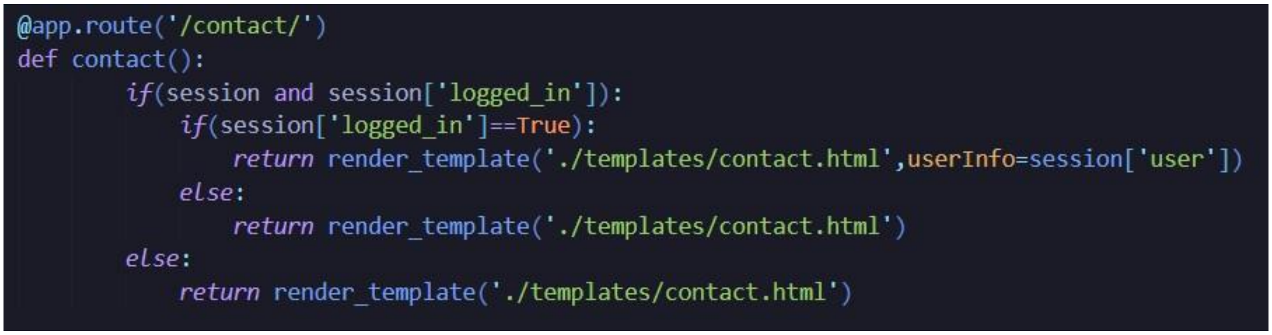
7.3 Feature 2

HISTORY PAGE



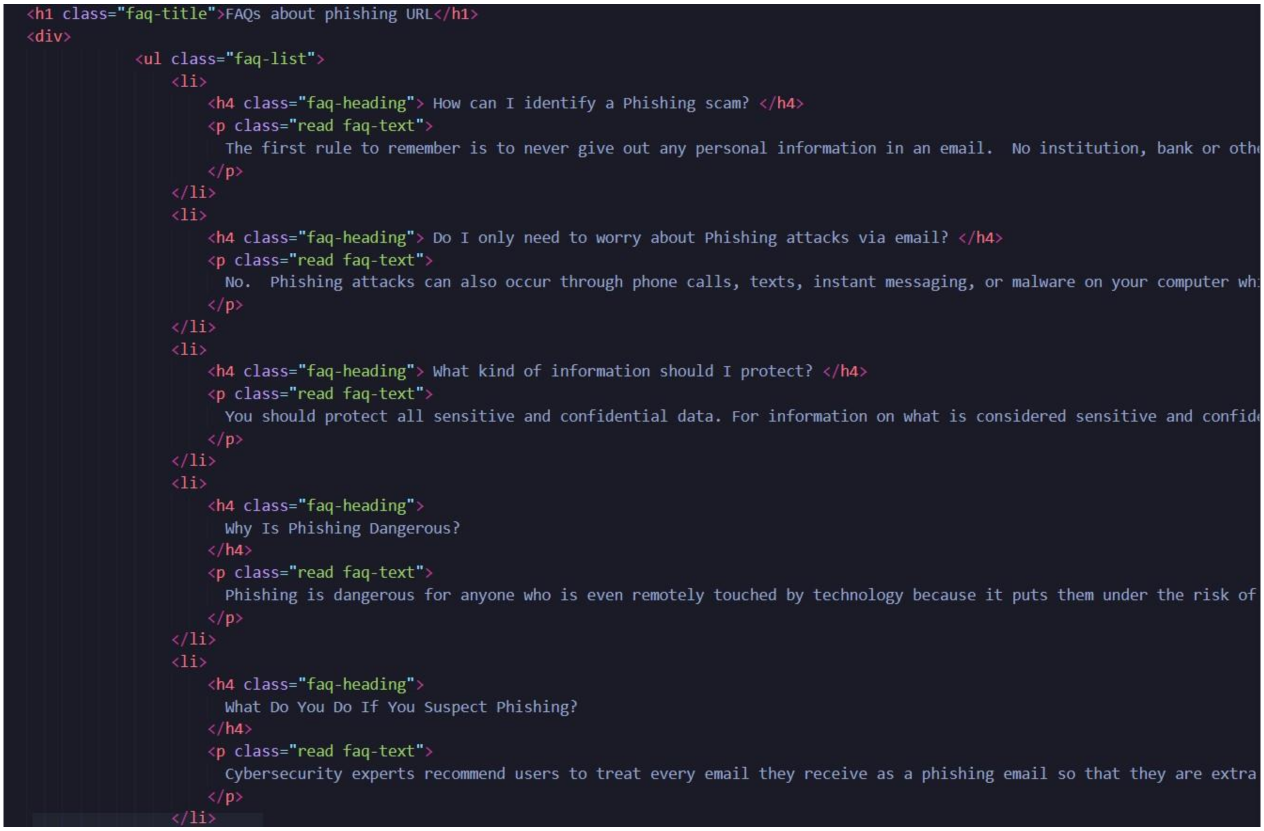
```python
@app.route('/detection-history/')
@login_required
def detectionHistory():
    if(session and session['logged_in']):
        if(session['logged_in']==True):
            get_detection_history_stmt = "SELECT title,url,status FROM detectionHistory where email=?"
            get_detection_history = ibm_db.prepare(conn, get_detection_history_stmt)
            ibm_db.bind_param(get_detection_history,1,session['user']['email'])
            ibm_db.execute(get_detection_history)
            fetch_detection_history = ibm_db.fetch_assoc(get_detection_history)
            detection_history = []
            ind = 0
            while fetch_detection_history != False:
                detection_history.append(fetch_detection_history)
                ind += 1
                fetch_detection_history = ibm_db.fetch_assoc(get_detection_history)
            detection_history= detection_history[::-1]
            return render_template('./templates/detection-history.html',userInfo=session['user'],detectionHistory=detection_history)
```

CONTACT US PAGE



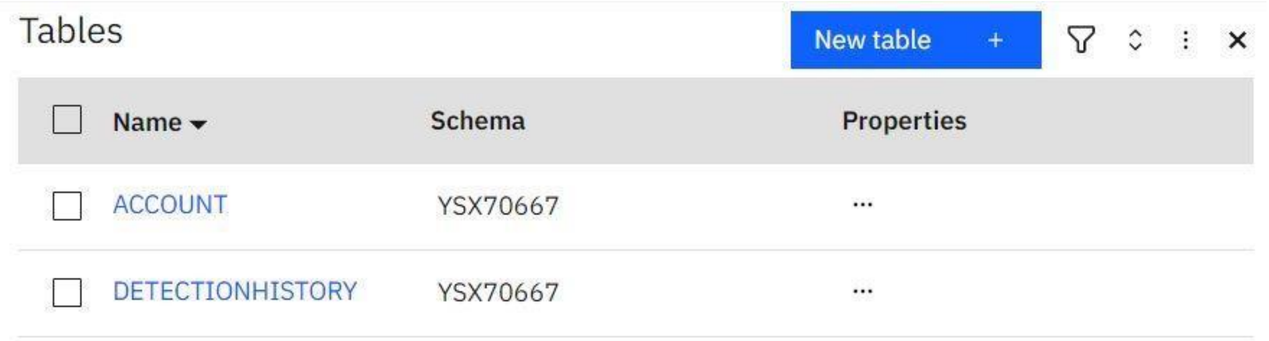
```python
@app.route('/contact/')
def contact():
        if(session and session['logged_in']):
            if(session['logged_in']==True):
                return render_template('./templates/contact.html',userInfo=session['user'])
            else:
                return render_template('./templates/contact.html')
        else:
            return render_template('./templates/contact.html')
```

FAQ



```html
<h1 class="faq-title">FAQs about phishing URL</h1>
<div>
            <ul class="faq-list">
                <li>
                    <h4 class="faq-heading"> How can I identify a Phishing scam? </h4>
                    <p class="read faq-text">
                      The first rule to remember is to never give out any personal information in an email.  No institution, bank or othe
                    </p>
                </li>
                <li>
                    <h4 class="faq-heading"> Do I only need to worry about Phishing attacks via email? </h4>
                    <p class="read faq-text">
                      No.  Phishing attacks can also occur through phone calls, texts, instant messaging, or malware on your computer wh
                    </p>
                </li>
                <li>
                    <h4 class="faq-heading"> What kind of information should I protect? </h4>
                    <p class="read faq-text">
                      You should protect all sensitive and confidential data. For information on what is considered sensitive and confid
                    </p>
                </li>
                <li>
                    <h4 class="faq-heading">
                      Why Is Phishing Dangerous?
                    </h4>
                    <p class="read faq-text">
                      Phishing is dangerous for anyone who is even remotely touched by technology because it puts them under the risk of
                    </p>
                </li>
                <li>
                    <h4 class="faq-heading">
                      What Do You Do If You Suspect Phishing?
                    </h4>
                    <p class="read faq-text">
                      Cybersecurity experts recommend users to treat every email they receive as a phishing email so that they are extra
                    </p>
                </li>
```

7.4 Database Schema (if Applicable)



□	Name ▾	Schema	Properties
□	ACCOUNT	YSX70667	...
□	DETECTIONHISTORY	YSX70667	...

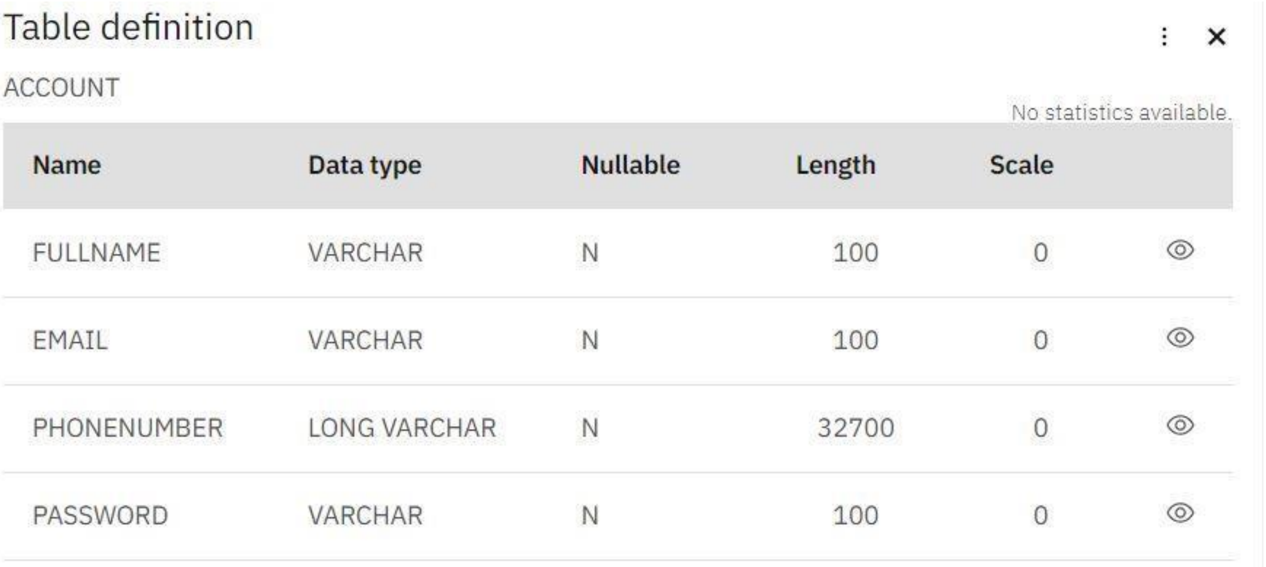


Table definition

ACCOUNT

No statistics available.

Name	Data type	Nullable	Length	Scale	
FULLNAME	VARCHAR	N	100	0	👁
EMAIL	VARCHAR	N	100	0	👁
PHONENUMBER	LONG VARCHAR	N	32700	0	👁
PASSWORD	VARCHAR	N	100	0	👁

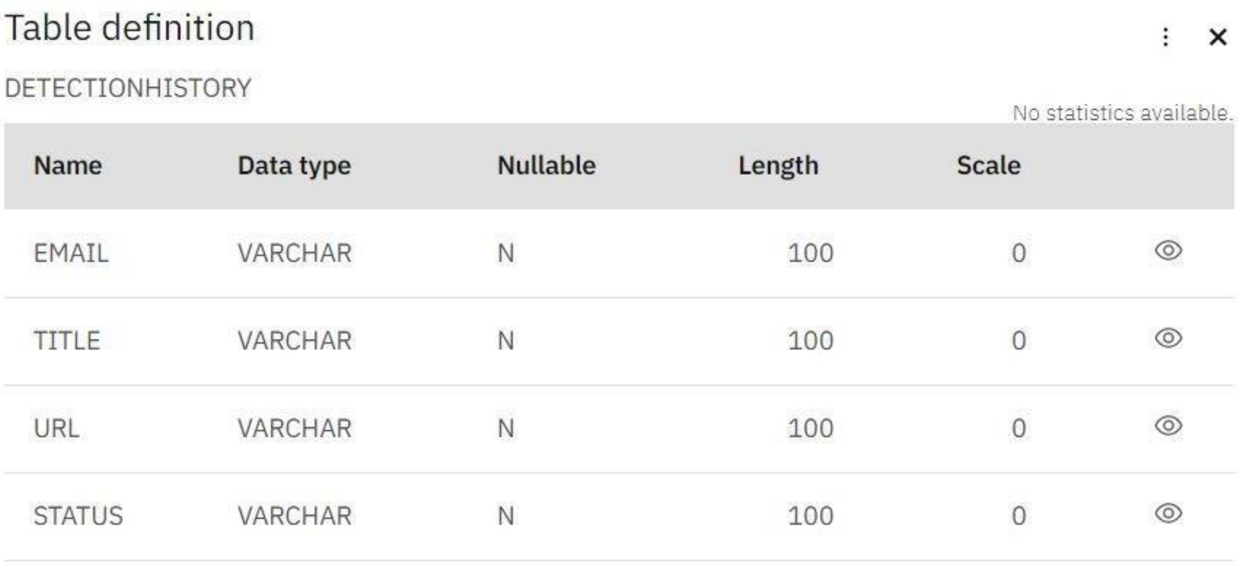


Table definition

DETECTIONHISTORY

No statistics available.

Name	Data type	Nullable	Length	Scale	
EMAIL	VARCHAR	N	100	0	👁
TITLE	VARCHAR	N	100	0	👁
URL	VARCHAR	N	100	0	👁
STATUS	VARCHAR	N	100	0	👁

8 TESTING

8.2 Test Cases

Test case ID	Feature Type	Component	Test Scenario	Pre-Requisite	Steps To Execute	Test Data	Expected Result	Actual Result	Status	Comments
Home page _TC_OO1	Functional	Home Page	Verify user is able to enter the URL in the form	Run the flask app in local host	1.Open our phishing website 2. Login to use the phishing services 3. Enter the link to be detected and click on predict button	https://google.com/	Result of classification will be displayed	Working as expected	Pass	Since www.google.com is a safe link, the output would display and say it is a safe link
Result Page _TC_OO1	UI	Contact us page	Verify the UI elements in the form	Run the flask app in local host	1. Enter name, email and message 2. Press submit	-	An email received stating that the message has been forwarded to the team	Working as expected	Pass	Email JS is used to send automatic email
Result Page _TC_OO2	Functional	Prediction result page	Verify user is able to see an alert when	Run the flask app in local host	1.Enter URL and click go		Alert of incomplete input	Working as expected	Pass	

			nothing is entered in the textbox		2.Enter nothing and click submit 3.An alert is displayed to provide proper input					
Prediction Page _TC_ OO1	Functional	Prediction form page	Verify user is able to see the result when URL is entered in the textbox	Run the flask app in local host	1.Enter URL and click go 2. Enter any URL and click submit 3. The result of the classification is displayed in a new page.	https://google.com/	Result of classification will be displayed with corresponding a emotion	Working as expected	Pass	

8.3 User Acceptance Testing

8.3.1 Defect Analysis:

This report shows the number of resolved or closed bugs at each severity level, and how they were resolved

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	10	4	2	3	20
Duplicate	1	0	3	0	4
External	2	3	0	1	6
Fixed	11	2	4	20	37
Not Reproduced	0	0	1	0	1
Skipped	0	0	1	1	2
Won't Fix	0	5	2	1	8
Totals	24	14	13	26	77

8.3.2 Test Case Analysis:

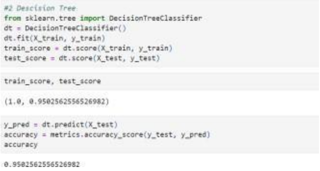
This report shows the number of test cases that have passed, failed, and untested

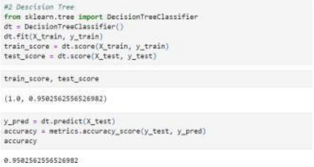
Section	Total Cases	Not Tested	Fail	Pass
Print Engine	5	0	0	5-
Client Application	51	0	0	51
Security	2	0	0	2
Outsource Shipping	3	0	0	3
Exception Reporting	9	0	0	9
Final Report Output	4	0	0	4
Version Control	2	0	0	2

9 RESULTS

9.2 Performance Metrics

S. No.	Parameter	Values	Screenshot

1.	Model Summary	**Decision Tree Model Accuracy –97%**	
			`#2 Descision Tree`
			`from sklearn.tree import DecisionTreeClassifier`
			`dt = DecisionTreeClassifier()`
			`dt.fit(X_train, y_train)`
			`train_score = dt.score(X_train, y_train)`
			`test_score = dt.score(X_test, y_test)`
			`train_score, test_score`
			`(1.0, 0.9502562556526982)`
			`y_pred = dt.predict(X_test)`
			`accuracy = metrics.accuracy_score(y_test, y_pred)`
			`accuracy`
			`0.9502562556526982`

2.	Accuracy	Training Accuracy -Test	
			`#2 Descision Tree`
			`from sklearn.tree import DecisionTreeClassifier`
			`dt = DecisionTreeClassifier()`
			`dt.fit(X_train, y_train)`
			`train_score = dt.score(X_train, y_train)`
			`test_score = dt.score(X_test, y_test)`
			`train_score, test_score`
			`(1.0, 0.9502562556526982)`
			`y_pred = dt.predict(X_test)`
			`accuracy = metrics.accuracy_score(y_test, y_pred)`
			`accuracy`
			`0.9502562556526982`

Model Performance Comparison:

https://dataplatform.cloud.ibm.com/analytics/notebooks/v2/8188710d-09bc-4dd6-824d-ec5519a26fea/view?access_token=c66af8d9cede342710725423df04821dc21cc53af0d4eae2f7496d9e1d3f5a7f

10 ADVANTAGES & DISADVANTAGES:

Phishing is the attempt to obtain a user's financial and personal information, such as credit card numbers and passwords, through electronic communication such as email and other messaging services. Attackers pose as representatives of a company and direct users to a fake website that looks like a phishing website, which is then used to gather personal data about users. A link embedded in the email can be used by attackers to trick users into downloading malware or malicious software.

To protect users from phishing attacks, numerous studies have been conducted. Firewalls, the blocking of specific domains and IP addresses, spam filtering methods, the detection of phoney websites, client-side toolbars, and user education are some of them. Both benefits and drawbacks may be seen in any of these methods now in use. The requirement to automatically identify phishing targets is a significant issue for anti-phishing initiatives. Knowing the website that is thought to be the target website allows us to identify which specific pages are phishing attempts. The owners may benefit from being able to recognize phishing attempts and take the appropriate countermeasures right away.

11 CONCLUSION

Using machine learning technologies, this initiative seeks to improve the detection process for phishing websites. Using the random forest approach, we had the lowest percentage of false positives and 97.14% detection accuracy. The outcome further demonstrates that classifiers perform better when more data is utilized as training data. Future phishing website detection will be more accurate thanks to the implementation of hybrid technology, which combines the blacklist approach with the random forest algorithm of machine learning.

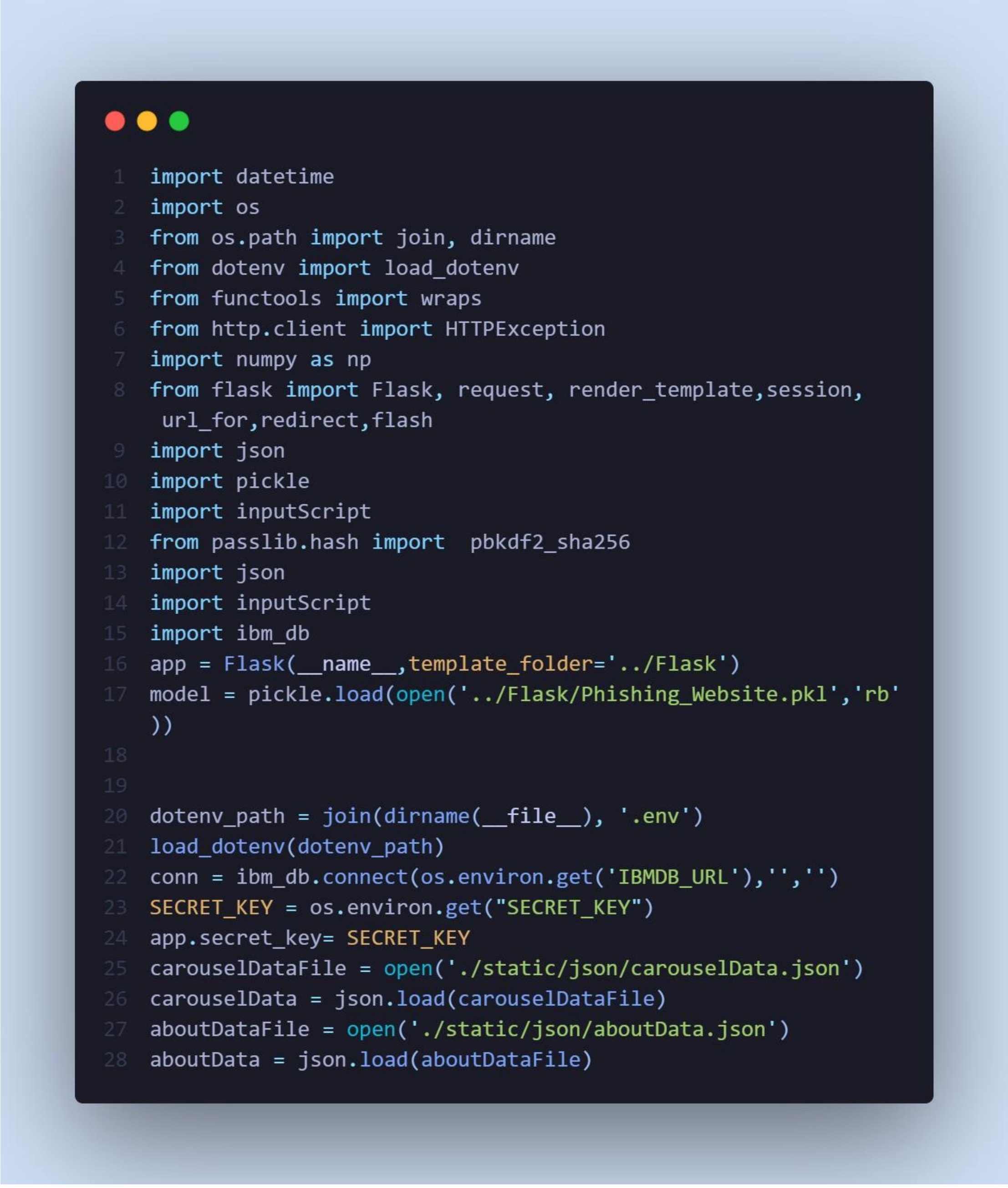
12 FUTURE SCOPE:

Future study will evaluate the effectiveness of the current finding with the use of a different method, such as deep learning, for phishing web page identification. Additionally, a web browser plug-in that can identify phishing websites and shield consumers in real time will be created based on an effective algorithm.

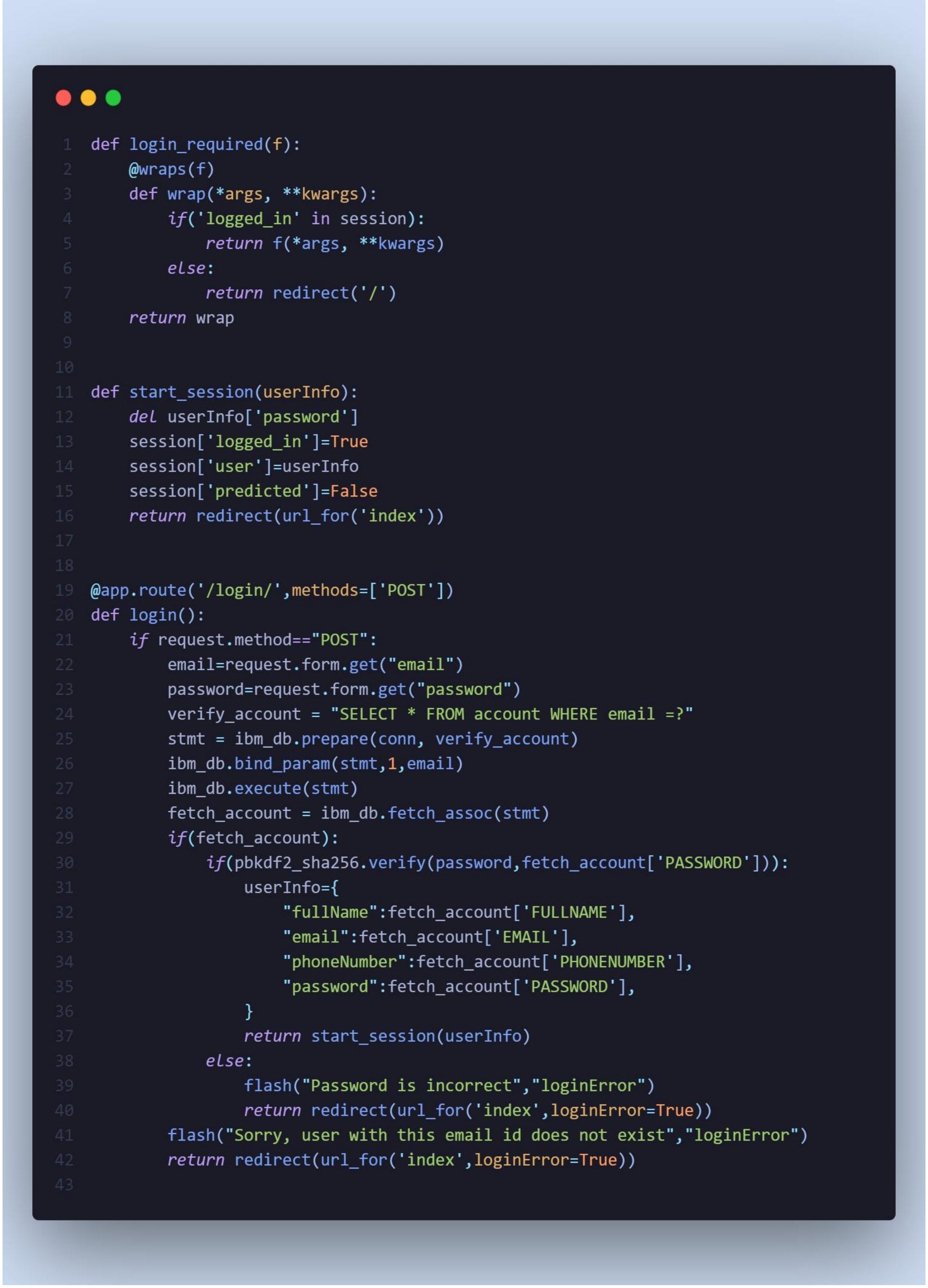
For simple access to human life, service providers provide a variety of the quickest instruments online. Additionally, online crime such as phishing is disseminated similarly to real-world crime. However, there is no online security team protecting users from these crimes. All types of internet users can benefit greatly from an anti-phishing program. These security tools are more necessary for beginners or people with limited internet or e-commerce knowledge. Phishing's primary targets are online banking or payments. The ideal method for identifying cybercrime or e-marketing fraud is thus an automated anti-phishing technique.

13 APPENDIX

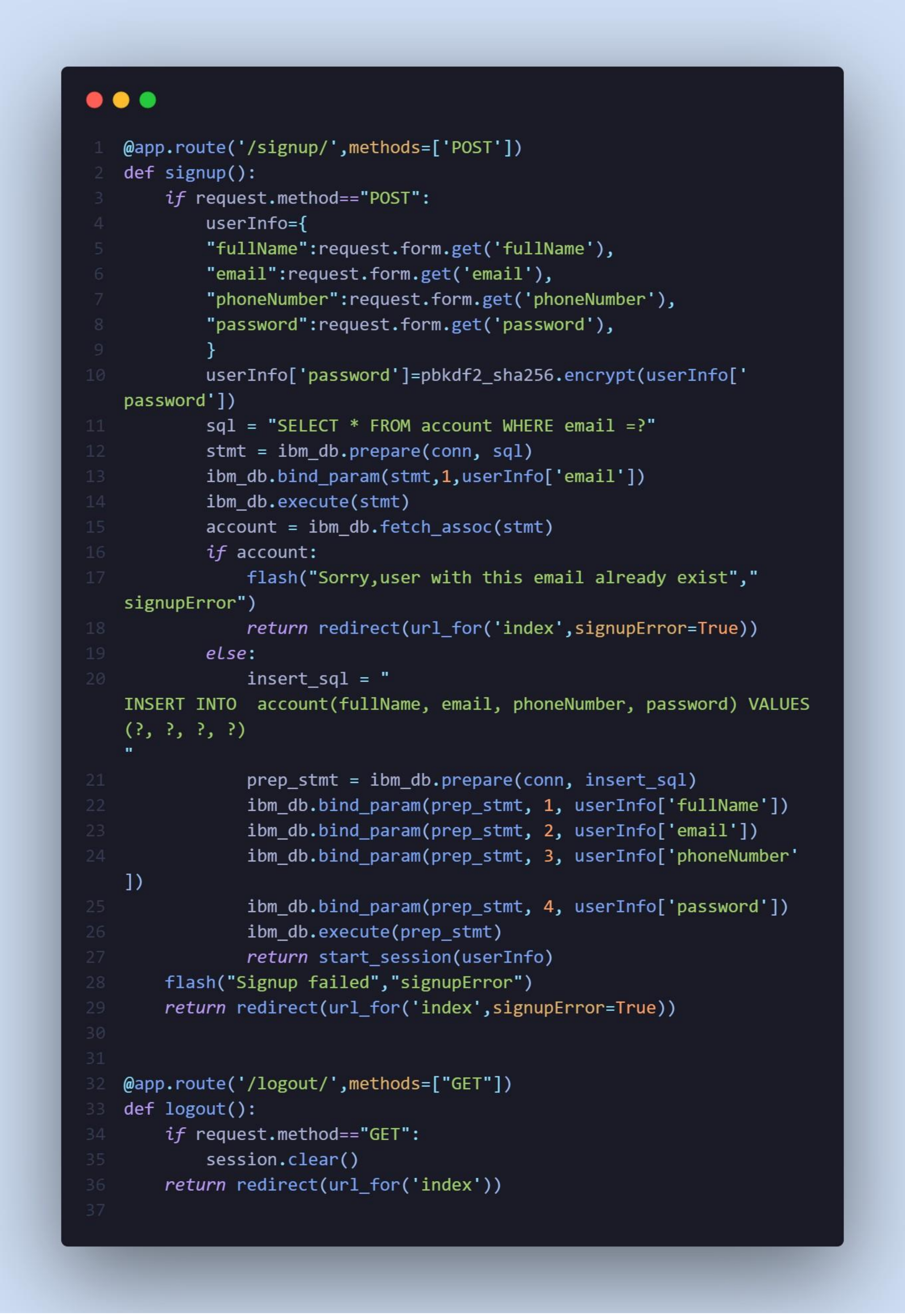
Source Code



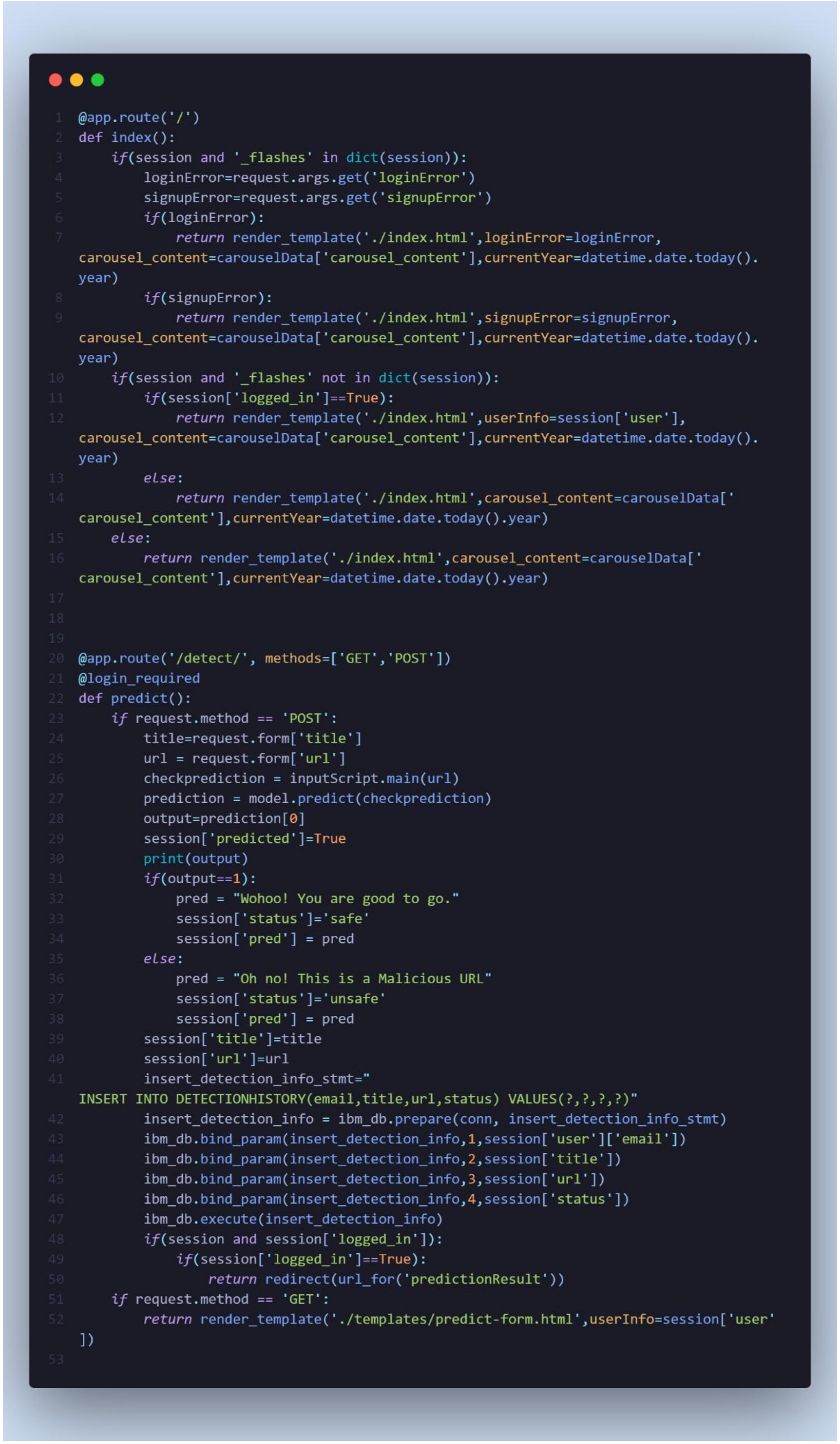
```python
import datetime
import os
from os.path import join, dirname
from dotenv import load_dotenv
from functools import wraps
from http.client import HTTPException
import numpy as np
from flask import Flask, request, render_template,session, url_for,redirect,flash
import json
import pickle
import inputScript
from passlib.hash import  pbkdf2_sha256
import json
import inputScript
import ibm_db
app = Flask(__name__,template_folder='../Flask')
model = pickle.load(open('../Flask/Phishing_Website.pkl','rb'))


dotenv_path = join(dirname(__file__), '.env')
load_dotenv(dotenv_path)
conn = ibm_db.connect(os.environ.get('IBMDB_URL'),'','')
SECRET_KEY = os.environ.get("SECRET_KEY")
app.secret_key= SECRET_KEY
carouselDataFile = open('./static/json/carouselData.json')
carouselData = json.load(carouselDataFile)
aboutDataFile = open('./static/json/aboutData.json')
aboutData = json.load(aboutDataFile)
```



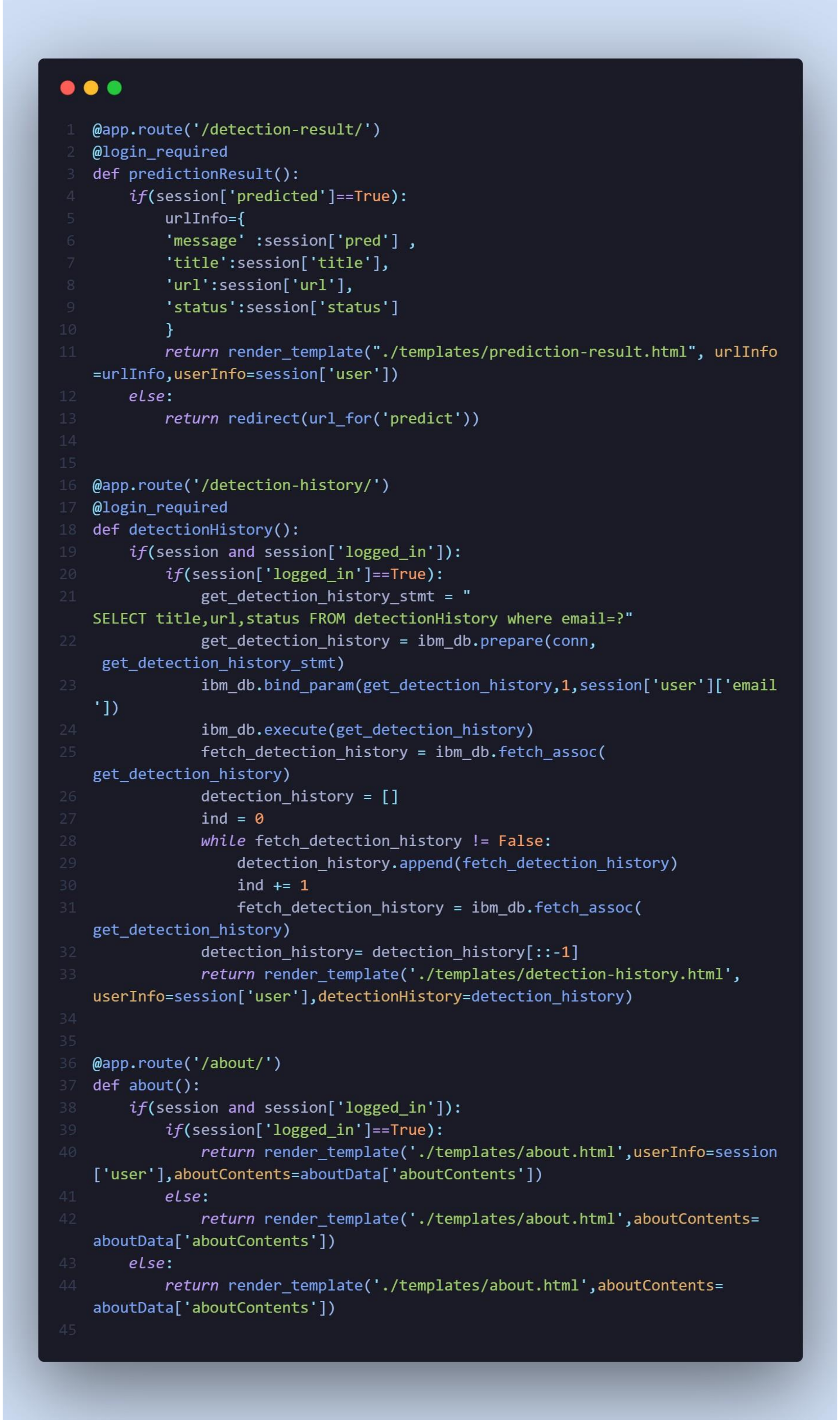
```python
def login_required(f):
    @wraps(f)
    def wrap(*args, **kwargs):
        if('logged_in' in session):
            return f(*args, **kwargs)
        else:
            return redirect('/')
    return wrap


def start_session(userInfo):
    del userInfo['password']
    session['logged_in']=True
    session['user']=userInfo
    session['predicted']=False
    return redirect(url_for('index'))


@app.route('/login/',methods=['POST'])
def login():
    if request.method=="POST":
        email=request.form.get("email")
        password=request.form.get("password")
        verify_account = "SELECT * FROM account WHERE email =?"
        stmt = ibm_db.prepare(conn, verify_account)
        ibm_db.bind_param(stmt,1,email)
        ibm_db.execute(stmt)
        fetch_account = ibm_db.fetch_assoc(stmt)
        if(fetch_account):
            if(pbkdf2_sha256.verify(password,fetch_account['PASSWORD'])):
                userInfo={
                    "fullName":fetch_account['FULLNAME'],
                    "email":fetch_account['EMAIL'],
                    "phoneNumber":fetch_account['PHONENUMBER'],
                    "password":fetch_account['PASSWORD'],
                }
                return start_session(userInfo)
            else:
                flash("Password is incorrect","loginError")
                return redirect(url_for('index',loginError=True))
        flash("Sorry, user with this email id does not exist","loginError")
        return redirect(url_for('index',loginError=True))
```



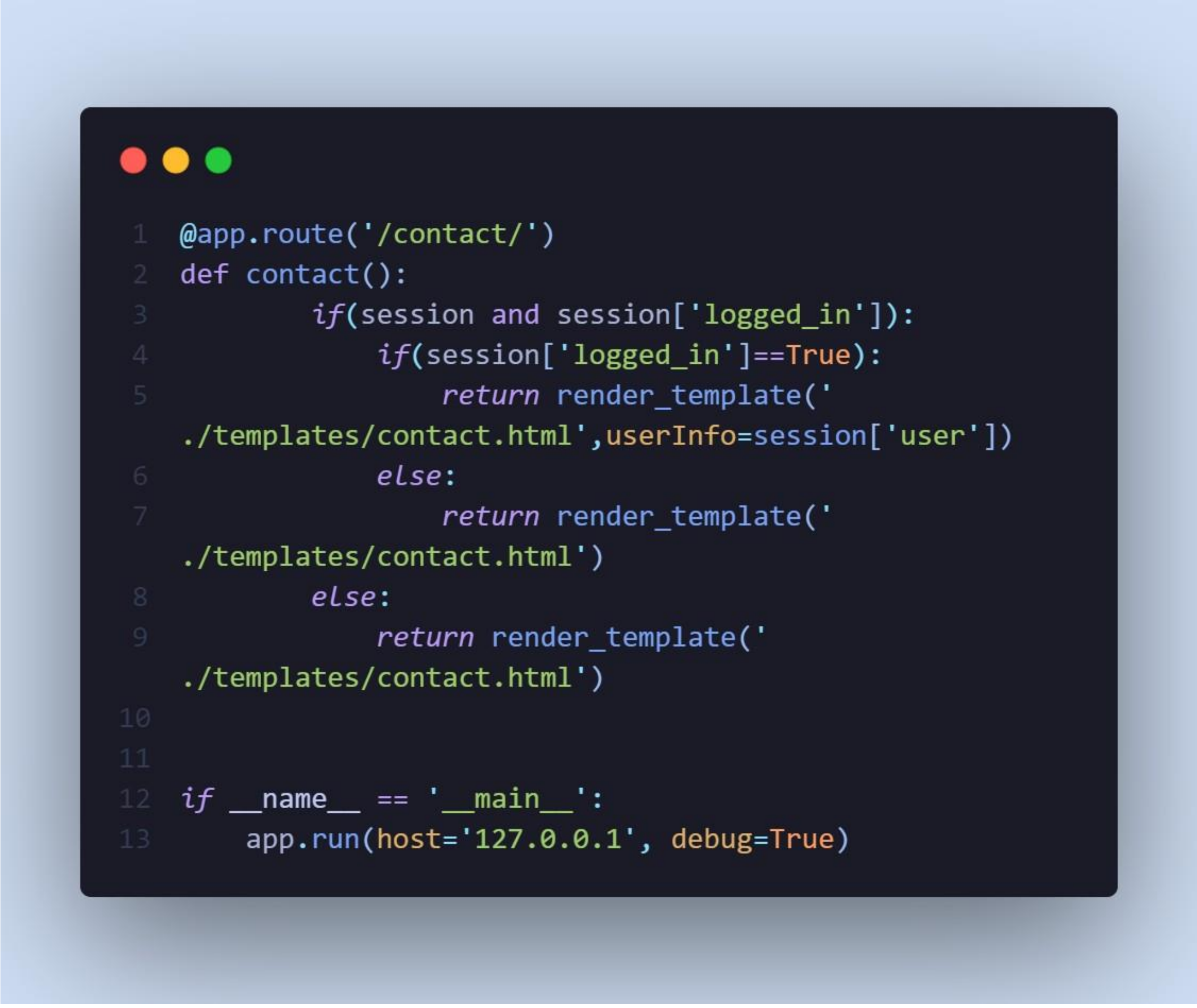
```python
@app.route('/signup/',methods=['POST'])
def signup():
    if request.method=="POST":
        userInfo={
        "fullName":request.form.get('fullName'),
        "email":request.form.get('email'),
        "phoneNumber":request.form.get('phoneNumber'),
        "password":request.form.get('password'),
        }
        userInfo['password']=pbkdf2_sha256.encrypt(userInfo['password'])
        sql = "SELECT * FROM account WHERE email =?"
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt,1,userInfo['email'])
        ibm_db.execute(stmt)
        account = ibm_db.fetch_assoc(stmt)
        if account:
            flash("Sorry,user with this email already exist","signupError")
            return redirect(url_for('index',signupError=True))
        else:
            insert_sql = "INSERT INTO  account(fullName, email, phoneNumber, password) VALUES (?, ?, ?, ?)"
            prep_stmt = ibm_db.prepare(conn, insert_sql)
            ibm_db.bind_param(prep_stmt, 1, userInfo['fullName'])
            ibm_db.bind_param(prep_stmt, 2, userInfo['email'])
            ibm_db.bind_param(prep_stmt, 3, userInfo['phoneNumber'])
            ibm_db.bind_param(prep_stmt, 4, userInfo['password'])
            ibm_db.execute(prep_stmt)
            return start_session(userInfo)
    flash("Signup failed","signupError")
    return redirect(url_for('index',signupError=True))


@app.route('/logout/',methods=["GET"])
def logout():
    if request.method=="GET":
        session.clear()
    return redirect(url_for('index'))

```



```python
@app.route('/')
def index():
    if(session and '_flashes' in dict(session)):
        loginError=request.args.get('loginError')
        signupError=request.args.get('signupError')
        if(loginError):
            return render_template('./index.html',loginError=loginError,carousel_content=carouselData['carousel_content'],currentYear=datetime.date.today().year)
        if(signupError):
            return render_template('./index.html',signupError=signupError,carousel_content=carouselData['carousel_content'],currentYear=datetime.date.today().year)
    if(session and '_flashes' not in dict(session)):
        if(session['logged_in']==True):
            return render_template('./index.html',userInfo=session['user'],carousel_content=carouselData['carousel_content'],currentYear=datetime.date.today().year)
        else:
            return render_template('./index.html',carousel_content=carouselData['carousel_content'],currentYear=datetime.date.today().year)
    else:
        return render_template('./index.html',carousel_content=carouselData['carousel_content'],currentYear=datetime.date.today().year)



@app.route('/detect/', methods=['GET','POST'])
@login_required
def predict():
    if request.method == 'POST':
        title=request.form['title']
        url = request.form['url']
        checkprediction = inputScript.main(url)
        prediction = model.predict(checkprediction)
        output=prediction[0]
        session['predicted']=True
        print(output)
        if(output==1):
            pred = "Wohoo! You are good to go."
            session['status']='safe'
            session['pred'] = pred
        else:
            pred = "Oh no! This is a Malicious URL"
            session['status']='unsafe'
            session['pred'] = pred
        session['title']=title
        session['url']=url
        insert_detection_info_stmt="INSERT INTO DETECTIONHISTORY(email,title,url,status) VALUES(?,?,?,?)"
        insert_detection_info = ibm_db.prepare(conn, insert_detection_info_stmt)
        ibm_db.bind_param(insert_detection_info,1,session['user']['email'])
        ibm_db.bind_param(insert_detection_info,2,session['title'])
        ibm_db.bind_param(insert_detection_info,3,session['url'])
        ibm_db.bind_param(insert_detection_info,4,session['status'])
        ibm_db.execute(insert_detection_info)
        if(session and session['logged_in']):
            if(session['logged_in']==True):
                return redirect(url_for('predictionResult'))
    if request.method == 'GET':
        return render_template('./templates/predict-form.html',userInfo=session['user'])
```



```python
@app.route('/detection-result/')
@login_required
def predictionResult():
    if(session['predicted']==True):
        urlInfo={
        'message' :session['pred'] ,
        'title':session['title'],
        'url':session['url'],
        'status':session['status']
        }
        return render_template("./templates/prediction-result.html", urlInfo=urlInfo,userInfo=session['user'])
    else:
        return redirect(url_for('predict'))


@app.route('/detection-history/')
@login_required
def detectionHistory():
    if(session and session['logged_in']):
        if(session['logged_in']==True):
            get_detection_history_stmt = "SELECT title,url,status FROM detectionHistory where email=?"
            get_detection_history = ibm_db.prepare(conn, get_detection_history_stmt)
            ibm_db.bind_param(get_detection_history,1,session['user']['email'])
            ibm_db.execute(get_detection_history)
            fetch_detection_history = ibm_db.fetch_assoc(get_detection_history)
            detection_history = []
            ind = 0
            while fetch_detection_history != False:
                detection_history.append(fetch_detection_history)
                ind += 1
                fetch_detection_history = ibm_db.fetch_assoc(get_detection_history)
            detection_history= detection_history[::-1]
            return render_template('./templates/detection-history.html', userInfo=session['user'],detectionHistory=detection_history)


@app.route('/about/')
def about():
    if(session and session['logged_in']):
        if(session['logged_in']==True):
            return render_template('./templates/about.html',userInfo=session['user'],aboutContents=aboutData['aboutContents'])
        else:
            return render_template('./templates/about.html',aboutContents=aboutData['aboutContents'])
    else:
        return render_template('./templates/about.html',aboutContents=aboutData['aboutContents'])
```



```python
@app.route('/contact/')
def contact():
        if(session and session['logged_in']):
            if(session['logged_in']==True):
                return render_template('./templates/contact.html',userInfo=session['user'])
            else:
                return render_template('./templates/contact.html')
        else:
            return render_template('./templates/contact.html')


if __name__ == '__main__':
    app.run(host='127.0.0.1', debug=True)
```

GitHub & Project Demo Link

GitHub: https://github.com/IBM-EPBL/IBM-Project-5336-1658758088

Demo Link: https://drive.google.com/file/d/1p6yzImA_48E6W5nJtrhyiCrtLx5kkPo3/view?usp=share_link